Blat User Manual

Compiled By Mark E. Donaldson

Description

Blat is a Public Domain Win32 console utility that sends the contents of a file in an e-mail message using the SMTP protocol. Blat is useful for creating scripts where mail has to be sent automatically. To use Blat you must have access to a SMTP server via TCP-IP. Blat can store various default settings in the registry. The command line options override the registry settings. Input from the console (STDIN) can be used instead of a disk file (if the special filename '-' is specified). Blat can also "carbon copy" and "blind carbon copy" the message. Impersonation can be done with the -i flag which puts the value specified in the "From:" line, however when this is done the real senders address is stamped in the "Reply-To:" and "Sender:" lines. This feature can be useful when using the program to send messages from users that are not registered on the SMTP host. Optionally, Blat can also attach multiple files to your message.

Installation

If you are upgrading from version 1.2 or later, simply copy Blat.exe over the old one. Blat no longer needs gensock.dll or gwinsock.dll. You can delete these unless another application you use requires them. If you are upgrading from Blat 1.1 or 1.0, or you never used Blat before you can copy the file "Blat.exe" to your "\WINNT\SYSTEM32" directory, or to any other directory in your path, or optionaly, run "Blat -install smtp.yoursite.tld youruserid@yoursite.tld"

Syntax

```
Blat <filename> -to <recipient> [optional switches (see below)]
  Blat -install <server addr> <sender's addr> [<try>[<port>[<profile>]]] [-q]
  Blat -profile [-delete | "<default>"] [profile1] [profileN] [-q]
  Blat -h

-install[SMTP|NNTP|POP3] <server addr> <sender's email addr> [<try n times>
[<port> [<profile> [<username> [<password>]]]]]
```

- set server, sender, number of tries and port for profile (<try n times> and <port> may be replaced by '-')
- port defaults are SMTP=25, NNTP=119, POP3=110
- default profile can be specified with a '-'
- username and/or password may be stored to the registry
- order of options is specific
- use -installNNTP for storing NNTP information
- use -installPOP3 for storing POP3 information (sender and try are ignored, use '-' in place of these)

```
blat.exe -install <your.mailserver.com> <you@yourdomain.com>
```

substituting your.mailserver.com and you@yourdomain.com with the name of your SMTP email server and your email address, respectively.

Installing it this way saves these values in your registry so you don't have to pass them as parameters every time you run Blat.

Basics
<filename>: file with the message body to be sent

- if your message body is on the command line, use a hyphen (-) as your first argument, and -body followed by your message
- if your message will come from the console/keyboard, use the hyphen as your first argument, but do not use -body option.

-of <file>: text file containing more options (also -optionfile)

-to <recipient>: recipient list (also -t) (comma separated)

-tf <file>: recipient list filename

-cc <recipient>: carbon copy recipient list (also -c) (comma separated)

-cf <file>: cc recipient list filename

-bcc <recipient>: blind carbon copy recipient list (also -b) (comma separated)

-bf <file>: bcc recipient list filename

-maxNames <x>: send to groups of <x> number of recipients

-ur: set To: header to Undisclosed Recipients if not using the -to and -cc options

-subject <subj>: subject line, surround with quotes to include spaces(also -s)

-ss: suppress subject line if not defined

-sf <file>: file containing subject line

-body <text>: message body, surround with quotes to include spaces

-sig <file>: text file containing your email signature

-tag <file>: text file containing taglines, to be randomly chosen

-ps <file>: final message text, possibly for unsubscribe instructions

Registry Overrides
-p <profile>: send with server, user, and port defined in <profile>: use username and password if defined in <profile>

-profile: list all profiles in the Registry

-server <addr>: specify SMTP server to be used (optionally, addr:port)

-serverSMTP <addr>: same as -server

-serverNNTP <addr>: specify NNTP server to be used (optionally, addr:port)

-serverPOP3 <addr>: specify POP3 server to be used (optionally, addr:port) when POP3 access is required before sending email

-f <sender>: override the default sender address (must be known to server)

-i <addr>: a 'From:' address, not necessarily known to the server

-port <port>: port to be used on the SMTP server, defaults to SMTP (25)

-portSMTP <port>: same as -port

-portNNTP <port>: port to be used on the NNTP server, defaults to NNTP (119)

-portPOP3 <port>: port to be used on the POP3 server, defaults to POP3 (110)

-u <username>: username for AUTH LOGIN (use with -pw)

-pw <password>: password for AUTH LOGIN (use with -u)

-pu <username>: username for POP3 LOGIN (use with -ppw)

-ppw <password>: password for POP3 LOGIN (use with -pu)

Miscellaneous RFC header Switches

-organization <organization>: Organization field (also -o and -org)

-ua: include User-Agent header line instead of X-Mailer

-x <X-Header: detail>: custom 'X-' header. eg: -x "X-INFO: Blat is Great!"

-noh: prevent X-Mailer/User-Agent header from showing Blat homepage

-noh2: prevent X-Mailer header entirely

-d: request disposition notification

-r: request return receipt

-charset <cs>: user defined charset. The default is ISO-8859-1

-a1 <header>: add custom header line at the end of the regular headers

-a2 <header>: same as -a1, for a second custom header line

-dsn <nsfd>: use Delivery Status Notifications (RFC 3461) n = never, s = successful, f = failure, d = delayed can be used together, however N takes precedence

-hdrencb: use base64 for encoding headers, if necessary

-hdrencq: use quoted-printable for encoding headers, if necessary

-priority <pr>: set message priority 0 for low, 1 for high

Attachment and Encoding Options
-attach <file>: attach binary file(s) to message (filenames comma separated)

-attacht <file>: attach text file(s) to message (filenames comma separated)

-attachi <file>: attach text file(s) as INLINE (filenames comma separated)

-embed <file>: embed file(s) in HTML. Object tag in HTML must specify content-id using cid: tag. eg: <img src="cid:image.jpg">

-af <file>: file containing list of binary file(s) to attach (comma separated)

-atf <file>: file containing list of text file(s) to attach (comma separated)

-aef <file>: file containing list of embed file(s) to attach (comma separated)

-enrich-base64: send binary files using base64 (binary MIME)

-uuencodeed: send binary files UUEncoded

-unicode: message body is in 16- or 32-bit Unicode format

-html: send an HTML message (Content-Type=text/html)

-alttext <text> : plain text for use as alternate text

-alttextf <file>: plain text file for use as alternate text

-mime: MIME Quoted-Printable Content-Transfer-Encoding

-8bitmime: ask for 8bit data support when sending MIME

-multipart <size>: send multipart messages, breaking attachments on <size> KB boundaries, where <size> is per 1000 bytes

-nomps: do not allow multipart messages

NNTP Specific Options
-groups <usenet groups>: list of newsgroups (comma separated)

Other Options
-h: displays this help (also -?, /?, -help or /help)

-q: suppresses all output to the screen

-debug: echoes server communications to a log file or screen (overrides -q if echoes to the screen)

-log <file>: log everything but usage to <file>

-timestamp: when -log is used, a timestamp is added to each log line

-ti <n>: set timeout to 'n' seconds. Blat will wait 'n' seconds for
<n times>: how many times blat should try to send (1 to 'INFINITE')

-try y: do not convert ASCII | (pipe, 0x7c) to CrLf in the message body

-hostname <hst>: select the hostname used to send the message via SMTP

-raw <x>:: hex/ascii dump the data between Blat and the server

-delay: wait x seconds between messages being sent when used with -maxnames or -multipart

-comment <char>: use this character to mark the start of commments in

-superdebugT: ascii dump the data between Blat and the server options files and recipient list files.

The default is -superdebug

Note that if the '-i' option is used, <sender> is included in 'Reply-to:' and 'Sender:' fields in the header of the message. Optionally, the following options can be used instead of the -f and -I options:

-mailfrom <addr> The RFC 821 MAIL From: statement
-from <addr> The RFC 822 From: statement
-replyto <addr> The RFC 822 Reply-To: statement
-returnpath <addr> The RFC 822 Return-Path: statement
-sender <addr> The RFC 822 Sender: statement

For backward consistency, the -f and -i options have precedence over these RFC 822 defined options. If both -f and -i options are omitted then the RFC 821 MAIL FROM statement will be defaulted to use the installation-defined default sender address.

EXAMPLE: Blat run from a Batch file

```
@echo off
:::::::::::::: Lets set some variables ::::::::::::::
set eMail=tim@blat.tld
```

```
set subj=-s "Test Blat"
set server=-server localhost
set x=-x "X-Header-Test: Can Blat do it? Yes it Can!"
set debug=-debug -log blat.log -timestamp
:::::::::::::::: Now we run Blat!  ::::::::::::::::
blat %0 -to %eMail% -f %eMail% %subj% %server% %debug% %x%
```

The above resolves to:

```
Blat batch.bat -to tim@blat.tld -f tim@blat.tld -s "Test Blat" -server localhost -
debug -log blat.log -timestamp -x "X-Header-Test: Can Blat do it? Yes it Can!"
```

EXAMPLE: Checking ErrorLevels Returned From Blat

```
@echo off
:::::::::::::: Lets set some variables ::::::::::::::
set email=tim@blat.tld
set server=-server localhost
set subject=-s "Test Blat ERRORLEVEL's"
set tof=-to %email% -f %email%
set msg=%0
:::::::::::::::: Now we run Blat!  ::::::::::::::::
blat %msg% %tof% %subject% %server%
:::::::::::: A quick message to the screen ::::::::::
echo.
echo ErrorLevel returned from Blat == %ERRORLEVEL%
echo.

:: Check each errorlevel from 0 through 14 in that order
set ELmsg=Blat returned this ERRORLEVEL
:::::::::::::: Here is the FOR loop! ::::::::::::::
FOR /L %%i IN (0,1,14) DO if ERRORLEVEL %%i echo %ELmsg% %%i
```

Here's an example of how you could capture the return codes from Blat. Important things to note is that error level checking is done beginning with the highest number, then going down to the lowest number. A zero means no error. Any additional commands to be executed if Blat runs successfully should follow the :NOERROR label. If an error does occur, the batch file will display an error message and then stop, waiting for the operator to press a key for it to then continue. If Blat runs with no errors (error level zero) then the batch file exits without any messages and without stopping.

```
:: Sample batch file for Blat
@ECHO OFF
blat (...rest of command line goes here)
IF ERRORLEVEL=13 THEN GOTO :ERROR13
IF ERRORLEVEL=12 THEN GOTO :ERROR12
IF ERRORLEVEL=5  THEN GOTO :ERROR5
IF ERRORLEVEL=4  THEN GOTO :ERROR4
IF ERRORLEVEL=3  THEN GOTO :ERROR3
IF ERRORLEVEL=2  THEN GOTO :ERROR2
IF ERRORLEVEL=1  THEN GOTO :ERROR1
:: If we get this far, then there was no error code
GOTO :NOERROR
```

```
:ERROR13
ECHO ERROR: Error opening temporary file in temp directory!
GOTO :EXIT

:ERROR12
ECHO ERROR: -server or -f options not specified or not found in registry
GOTO :EXIT

:ERROR5
ECHO ERROR: Error reading file/message text
GOTO :EXIT

:ERROR4
ECHO ERROR: Problem with the file/message text
GOTO :EXIT

:ERROR3
ECHO ERROR: Error reading file/message text or attached file
GOTO :EXIT

:ERROR2
ECHO ERROR: Error Level 2 returned
GOTO :EXIT

:ERROR1
ECHO ERROR: Error Level 1 returned
GOTO :EXIT

:NOERROR
:: Any additional batch processing goes here
GOTO :EXITNOPAUSE

:EXIT
PAUSE
:EXITNOPAUSE
:: End of batch file here
```

EXAMPLE: Blat run from Perl

```
#!perl
use strict;
use warnings;

my $Debug=0; # =0 gets NO console output
my ($DTS, $SMTP, $subject, $body);
DTS(); # go build the Date Time Stamp

$SMTP="localhost"; # the SMTP mail server name or IP address

$subject="\"Testing blat $DTS\"";

my %Envelope = ( # Envelope bits
    -f  => 'tim@blat.tld', # FROM:
    -to => 'tim@blat.tld', # TO:
```

```
);

my %Data = ( # Data bits
        # you can add other parms here as needed
    -subject  => $subject,   # just what it sounds like!
    -server   => $SMTP,      # specify the SMTP server to use
    -debug    => " ",        # run Blat with debuging output
    -log      => "\"$0.log\"", # dump screen output to a file instead.
);

GetBodyFromFile($0); # go get the message body from a file
# put "D:/path/filename.txt" in place of $0 to use your own file
# $body = 'You can also set the $body Variable directly if you don\'t want to get
the message body from a file';

SendIt($body, %Envelope, %Data);

sub SendIt {
        my $BlatCmd="Blat.exe"; # the Blat binary (path if needed)

        my $body = shift @_; # get the msg body
        $BlatCmd .= " - @_"; # add all the parms
        # now we have something like
        # blat.exe - -f tim@blat.tld -to tim@blat.tdl
        #          -debug -log "C:\mail\tims-blat.pl.log"
        #          -server localhost
        #          -subject "Testing blat 2003-8-20 12:46:6"

        print "\n\nCommand Line = $BlatCmd\n\nMessage Body = $body\n" if $Debug;

        open (MAIL, "| $BlatCmd") || die $!; # start Blat with all it's parms
        print MAIL $body; # now put in the msg body (bigger this way than CL)
}

sub DTS { # build the Date Time Stamp
        #Sec=$T[0],M=1,H=2, mDay=3,Mon=4,Yr=5, wDay=6,yDay=7, isDST=8
        my (@T)=localtime; ++$T[4]; $T[5]+=1900;
        $DTS = qq[$T[5]-$T[4]-$T[3] $T[2]:$T[1]:$T[0]];
}

sub GetBodyFromFile {
        my $file = shift @_;
        local $/; # set the 'input record separator' to nothing
        open INFO, $file or die "Cannot open $file: $!"; # open it
        $body = <INFO>; # slurp it into $body
}
```

Telent To Smtp Server

Command What we are doing telnet mail.blat.tld 25 telnet to a machine that responds to smtp on port 25 **HELO blat.tld** issue the HELO command to start. **mail from:tim@blat.tld** issue the mail from command **rcpt to:tim@blat.tld** issue the rcpt to command (this should be a valid address) **data** now the data command will allow you to do the rest.

I typed the stuff in **blue**.

"Subject:The text you want on the subject line" on the first line for a subject other headers and the message body follow end the mail message with a . on a line by itsel.

B:\>**telnet blat.tld 25**
220 miniRelay Server v0.9.75 ready
HELO blat.tld
250 Hello blat.tld
mail from:tim@blat.tld
250 tim@blat.tld Address Okay
rcpt to:tim@blat.tld
250 tim@blat.tld Address Okay
data
354 Start mail input; end with <CRLF>.<CRLF>
subject:The text you want on the subject line

This is the message body!
.
250 Ok
quit
221 Signing Off

How and Why to Run 'Blat -install'

Blat does not need to be installed!

All the 'Blat -install' command does is store various parameters in the Windows Regestry!
-install <server addr> <sender's addr> [<try n times> [<port> [<profile> [<username> [<password>]]]]]

Syntax	Note: the [] characters indicate optional parameters. However, you may not skip any!	
-install	the parameter indicating to Blat that the following are install parms :-)	
	<server addr>	SMTP Server DNS name or TCP/IP address
	<sender's addr>	This is the "from" address - usually like tim@blat.tld
	***[<try n times>**	How many times to try to send the message (defaults to 0)
	***[<port>**	What TCP/IP Port to connect to (defaults to 25/119)
	***[<profile>**	allows Blat to track multiple sets of settings (use -p to reference)
	[<username>	UserName used for Authentication
	[<password>]]]]]	Password used for Authentication
	[-q]	Tell Blat to be quite when storing install settings in the registry
	* may be replaced by '-' to set default, and allow following options port defaults are SMTP=25, NNTP=119 order of options is specific use -installNNTP for storing NNTP information	
Examples		

Blat -install	**Sets the following** (& = cumulative with the previous line)
blat -install localhost	server='localhost'
blat -install localhost tim@blat.tld	& from address='tim@blat.tld'
blat -install localhost tim@blat.tld 3	& number of times to send the message = 3
blat -install localhost tim@blat.tld 3 26	& TCP/IP Port to send the mail using = 26
blat -install localhost tim@blat.tld 3 26 FromTim1	& store the settings in a Profiles named : FromTim1
blat -install localhost tim@blat.tld - - FromTim2	use the default number of tries, and Port!

Blat Return Codes

Code	Description
2	• The server actively denied our connection. • The mail server doesn't like the sender name.
1	• Unable to open SMTP socket • SMTP get line did not return 220 • command unable to write to socket • Server does not like To: address • Mail server error accepting message data.
0	OK
1	File name (message text) not given
1	Bad argument given
2	File (message text) does not exist
3	Error reading the file (message text) or attached file
4	File (message text) not of type FILE_TYPE_DISK
5	Error Reading File (message text)
12	-server or -f options not specified and not found in registry
13	Error opening temporary file in temp directory
Codes from /* $Id: gensock.h 1.8 1995/01/25 23:28:11 rushing Exp $ */	
4001	ERR_CANT_MALLOC
4002	ERR_SENDING_DATA
4003	ERR_INITIALIZING
4004	ERR_VER_NOT_SUPPORTED
4005	ERR_EINVAL
4006	ERR_SYS_NOT_READY
4007	ERR_CANT_RESOLVE_HOSTNAME
4008	ERR_CANT_GET_SOCKET
4009	ERR_READING_SOCKET
4010	ERR_NOT_A_SOCKET
4011	ERR_BUSY
4012	ERR_CLOSING
4013	WAIT_A_BIT

4014	ERR_CANT_RESOLVE_SERVICE
4015	ERR_CANT_CONNECT
4016	ERR_NOT_CONNECTED
4017	ERR_CONNECTION_REFUSED
-5000	ERR_NO_ERROR_CODE This is returned by misbehaving stacks that fail, but don't set an error code

Gensock Error Code Descriptions

- case 4001: printf("Error: Malloc failed (possibly out of memory)."); break;
- case 4002: printf("Error: Error sending data."); break;
- case 4003: printf("Error: Error initializing gensock.dll."); break;
- case 4004: printf("Error: Version not supported."); break;
- case 4005: printf("Error: The winsock version specified by gensock is not supported by this winsock.dll."); break;
- case 4006: printf("Error: Network not ready."); break;
- case 4007: printf("Error: Can't resolve (mailserver) hostname."); break;
- case 4008: printf("Error: Can't create a socket (too many simultaneous links?)"); break;
- case 4009: printf("Error: Error reading socket."); break;
- case 4010: printf("Error: Not a socket."); break;
- case 4011: printf("Error: Busy."); break;
- case 4012: printf("Error: Error closing socket."); break;
- case 4013: printf("Error: Wait a bit (possible timeout)."); break;
- case 4014: printf("Error: Can't resolve service."); break;
- case 4015: printf("Error: Can't connect to mailserver (timed out if winsock.dll error 10060)"); break;
- case 4016: printf("Error: Connection to mailserver was dropped."); break;
- case 4017: printf("Error: Mail server refused connection."); break;
- default: printf("error %d in function '%s'", retval, function);